51CTO.com
技术成就梦想
我们只谈开发
developer.51cto.com

开发月刊

Development Monthly

2011年4月
总第001期

走进对日外包
程序员的世界

四月编程语言
排行榜

开发者评判Android 3.0
优点与不足

目录 2011.04 contents

51CTO.com
技术成就梦想

出版方：北京无忧创想信息技术有限公司

责任编辑：彭凡

联系方法：pengfan@51cto.com 010-68476606（分机 8058）

出版日期：2011 年 4 月 14 日

每月 14 日出版

4月编程语言排行榜：告别Smalltalk

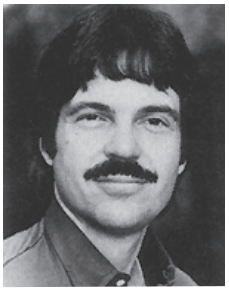
Tiobe 今天发布了 2011 年 4 月编程语言排行榜。令人叹息的是本期编程语言排行榜 Smalltalk 跌出前 50 名，Smalltalk 可是面向对象的程序设计语言的鼻祖，被公认为历史上第二个面向对象的程序设计语言，和第一个真正的集成开发环境 (IDE)。

4 月 4 日，Tiobe 发布了最新的编程语言排行榜。Java 与 C 语言继续占据头两位排名，值得一提的是 C# 升到第四，Lisp 由第 23 位升到 15 位，Ada 则由第 32 位跃居第 16 位，另外在过去的几月中，lua 的排名一直在上升。而且在 iPhone App Store 排名前十的应用都使用 Lua 来定义应用逻辑，包括极为火热的《愤怒的小鸟》。本期编程语言排行榜还是将会给大家介绍一门古典语言——Smalltalk。

以下是前 20 名编程语言排行：

Position Apr 2011	Position Apr 2010	Delta in Position	Programming Language
1	2	↑	Java
2	1	↓	C
3	3	=	C++
4	6	↑↑	C#
5	4	↓	PHP
6	7	↑	Python
7	5	↓↓	(Visual) Basic
8	11	↑↑↑	Objective-C
9	8	↓	Perl
10	10	=	JavaScript
11	12	↑	Ruby
12	20	↑↑↑↑↑↑↑↑	Lua
13	9	↓↓↓↓	Delphi
14	-	=	Assembly
15	23	↑↑↑↑↑↑↑↑	Lisp
16	32	↑↑↑↑↑↑↑↑↑↑	Ada
17	16	↓	Pascal
18	21	↑↑↑	Transact-SQL
19	-	=	Scheme
20	15	↓↓↓↓↓	Go

Smalltalk 由 Alan Kay，Dan Ingalls，Ted Kaehler，Adele Goldberg 等于 70 年代初在 Xerox PARC 开发，然而至今提起 Smalltalk 大家却不像对其他古老语言 Ada、Fortran、Lisp 那么陌生，这自然要归功于它的影响力。Smalltalk 可是面向对象的程序设计语言的鼻祖。它是纯面向对象的语言，就连整数也是对象，被公认为历史上第二个面向对象的程序设计语言，和第一个真正的集成开发环境 (IDE)。Smalltalk 对其它众多的程序设计语言的产生起到了极大的推动作用，主要有：Objective-C，Actor，Java 和 Ruby 等。90 年代的许多软件开发思想得利于 Smalltalk，例如设计模式、敏捷编程和重构等。Ward Cunningham，一位 Smalltalk 程序员发明了 WikiWiki。



Alan Kay

计算机学会 (ACM) 于 2004 年 4 月 19 日宣布，2003 年度有“计算机界诺贝尔奖”之称的 ACM 图灵奖授予第一个完全面向对象的动态计算机程序设计语言 Smalltalk 的发明者 Alan Kay。

4 月编程语言排行榜：告别 Smalltalk

Smalltalk 源自 Alan Kay “使用一组独立的互相通信的对象来解决问题”的思想，它可以说是目前主流语言 C++，Java 和 C# 的前身。也正是 Alan Kay 发明了“Object Oriented”这个术语。1972 年，他来到施乐 PARC，开始将 Smalltalk 作为一种儿童教育工具。在 PARC 期间他还与同事构建了最早的图形界面个人电脑 Alto，成为 Machitosh 和 Windows 的先驱。

Smalltalk 语法简单

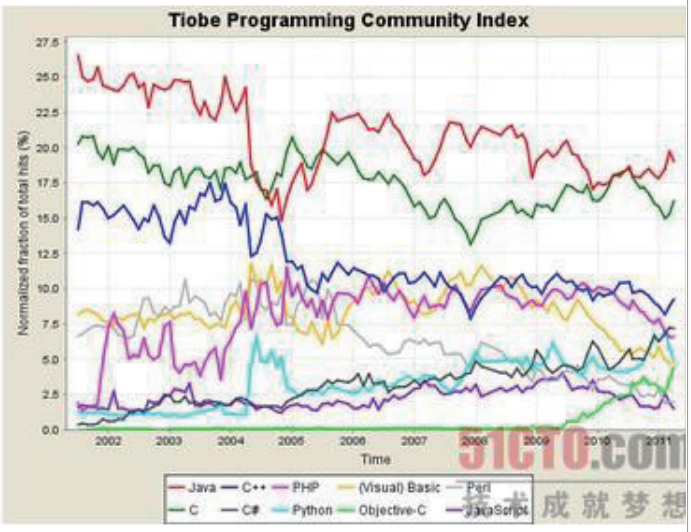
Smalltalk 语法简单到可以在 20 分钟内给你讲完它所有的语法。“它是完全面向对象的，debugger 允许“现场”调试，甚至是在一个 Web 环境中。Smalltalk 包含内建的源代码管理，而且可以直接与系统中的任何库交互，或通过 FFI 进行交互。Smalltalk 可以与文件、设备和 socket 交互，其能力丝毫不逊于任何 Perl 程序。而且，Smalltalk 已经历经长时间的检验。你与系统交互的任何一部分都是开放的，可编辑的，可订制的，包括你的开发工具在内。”

告别 Smalltalk

令人叹息的是本期编程语言排行榜 Smalltalk 跌出前 50 名，这款全球最古老的纯面向对象编程语言为众多新生代语言让开了道路。故在清明节将至，51CTO 给大家重点介绍 Smalltalk，不过我们相信 Smalltalk 只是短暂的离别，毕竟它是那么优秀，甚至一直被模仿从未被超越。

下面是本期编程语言排行榜的其他排名数据和趋势走向。

前 10 名编程语言走势图



下面是第 50 到 100 的编程语言排名

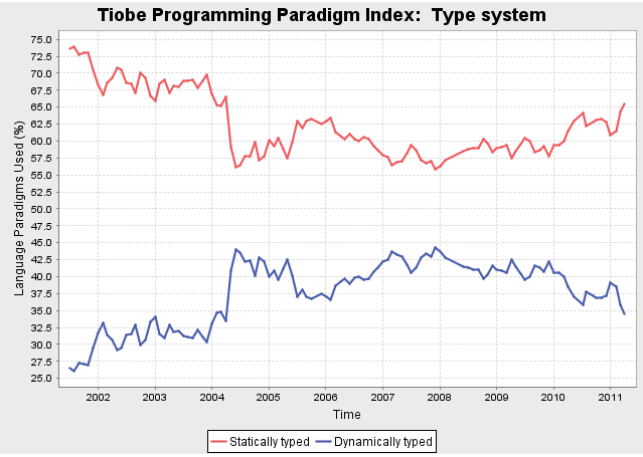
(Visual) FoxPro, ABC, Algol, Alpha, Arc, ATLAS, Avenue, Awk, Bash, bc,

Beta, Boo, Bourne shell, CFML, cg, CL (OS/400), Clean, cT,

Dylan, Eiffel, Factor, Groovy, Inform, Io, J, JavaFX Script, Korn shell, LabVIEW,

MAD, Magic, Maple, Mercury, MUMPS, NXT-G, Oberon, Object Pascal, Objective Caml, Occam, OpenCL, Oz, PILOT, PowerShell, Prolog, Revolution, S, Scala, Smalltalk, Spark, Standard ML, Tcl

下面给出了编程语言类别的一年变化趋势



对日软件外包相信大家并不陌生，但是这个行业的程序员究竟是怎样的一种状态？我们想通过他们自己的话来让大家有更深刻的了解。让我们一起走进对日外包程序员的世界，了解他们的技术人生。

专题链接：http://developer.51cto.com/art/201103/252468.htm

在这里我们选取了四位不同阶层的对日外包程序员作为我们的采访对象。他们中既有十年工作经验的项目经理，也有刚入行一年的初级程序员。不同的是他们的个人经历，相同的是他们对于日本软件开发行业严谨作风的领悟。这可能正是目前浮躁的中国 IT 业所缺失的重要素质。

三百六十行，行行出状元，对日外包程序员也有自己的春天。

51CTO 开发频道寄语

编者按

看过了很多人在对日外包行业成功或失败的经历，其实更多人想知道的是刚入行的“新兵”是怎么做的。由于是新入行的程序员，所以赵先生的个人感触会比较简短。

刚入行的对日外包程序员如是说

赵先生自嘲自己是从事对日外包工作的“新兵蛋子”，现在北京上地一家软件园外包企业工作。

51CTO：您从事对日外包已经几年？当初是怎么进入这一行的？

赵先生：已经从事对日外包 1.5 年，当初通过在培训机构学习进入对日外包这一行。

对日外包的程序员，很多都是从培训机构里入行。专门的培训机构将严格按照对日外包的要求，从语言到技术进行培训。由于是职业培训，所以没有大学里的理论教学，都是从工作出发。好的方面就是上手快，但是对于一个程序员更深层次的发展却是一种损害。很多人还没怎么学会走路，就要求他们跑起来。对程序开发还一知半懂，却要真的拿起工具来进行开发。

开发小 TIP

如果要从 PHP 数组中删除一个元素，请使用 unset() 函数，如：

```
unset($capitals['California']);
```

使用数字索引数组时，删除数组元素的办法更多，更灵活，可以使用 array_shift() 和 array_pop() 函数分别从数组的开头和末尾删除一个元素

51CTO：你平时使用的开发技术？目前您正在做哪方面的开发工作？

赵先生：目前在用 Java 做开发。主要也是这方面的项目。

51CTO：在这几年的工作经历中，您觉得在技术上提高的最大困难是什么？

赵先生：在于对式样的理解。特别是设计过程中的灵感问题。

51CTO：对于刚接触对日外包的新生，您有怎样的建议？

赵先生：做事要细心，认真。从每件事里琢磨出更多的东西来。举一反三的态度，可以让对日外包程序员学到更多。

51CTO：对日外包与国内的项目有什么区别？

赵先生：日本的正规，日本人做的东西每一次都是用心做的，比如网页上的一个按钮的样式，页面可以不好看，但是整体布局要合理，给你一看就是用心做的，还比如测试，一切用事实来说话，比如测试点，会反复的测试，等等……

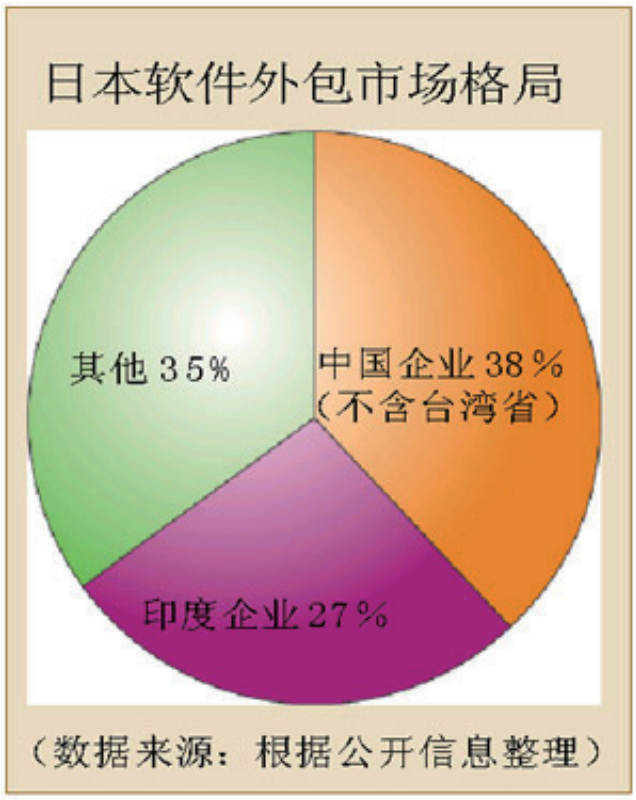
重视用户体验，为此进行反反复复的测试。而国内的很多项目目前对测试还是采取过于简单的方式，往往 BUG 频出，用户反响不好。

被逼上梁山的对日软件外包程序员

作者 / 彭凡

王先生工作在北京中关村上地软件园的一家外包公司，从事对日外包已经有四年的时间，由于公司业务调整，也曾做过国内政府的软件开发项目，对国内与对日项目有着很深刻的理解。

在走进王先生之前，我们先要了解目前中国对日软件外包的情况。虽然中国工资水平在世界处于靠后的位置，但还是受到类似印度这样的外包大国冲击，下图就是日本外包软件格局。我们可以看到中国只占到约三分之一份额，就是这三分之一的份额养活了一大批中国软件外包企业。



51CTO: 您从事对日外包已经几年？当初是怎么进入这一行的？

王先生：已经从事外包软件开发工作四年多了。当初毕业找不到工作，逼不得已才干外包的。

51CTO: 你平时使用的开发技术？目前您正在做哪方面的开发工作？

王先生：.NET、Java、Oracle、SQL Sever 都有过接触，目前做 .NET 方面的开发工作

由此我们可以看到软件外包的程序员完全是按照订单生产，各种语言都会涉及到。这就可能造成程序员样样会，但是样样都不精的局面。

51CTO: 在这几年的工作经历中，您觉得在技术上提高的最大困难是什么？

王先生：日语，作为日语是和客户交流的工具，所以日语不懂，就很容易跑偏。

实质上就是对需求的不明确造成的跑偏。由于语言上的障碍，无法直接交流而偏离了客户的设想。

51CTO: 对于刚接触对日外包的新生，您有怎样的建议？

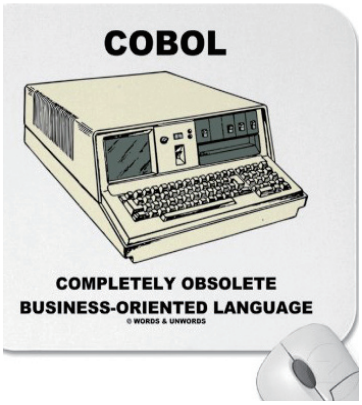
王先生：克服浮躁，脚踏实地。

既然进入了对日软件外包的行业，就安心工作。做好本质工作也能把自己的开发能力得到提高，只要不是安于现状就会有进步。

51CTO; 对日外包与国内的项目有什么区别？

王先生：各个方面要求很细，比如性能，代码的规范性，测试的密度。这也体现了日本人做事近乎变态的严谨性，就如同日本制造一样。

用COBOL的外包程序员



COBOL（Common Business Oriented Language）是数据处理领域最为广泛的程序设计语言，是第一个广泛使用的高级编程语言。于 1959 年由美国计算机用户组织设计了专用于商务处理的计算机语言 COBOL

李先生是在北京中关村上地软件园的一家外包公司工作的网友，去日本做过开发，喜欢在日本的生活，回到国内后也一直从事对日外包软件开发的工作。

对日外包开发分为驻日开发和在中国本土开发，很多人似乎很羡慕那些能到日本去工作的外包程序员。能接触到陌生的国家，还能看到更多更新的日本动漫。但是通过对李先生的采访，我们还是感受到即使身在日本，隔阂无处不在。比如日本人更喜欢通过靠邮件而不是电话沟通，日语的不熟练让外包程序员没多少机会跟日方接触。工作好几年后还只是简单的代码编写者，很难成长和晋升。

51CTO：您从事对日外包已经几年？当初是怎么进入这一行的？

李先生：6 年，大学学计算机专业

51CTO：你平时使用的开发技术？目前您正在做哪方面的开发工作？

李先生：COBOL；目前在做设计。

COBOL（COmmonBusinessOrientedLanguage）——是数据处理领域最为广泛的程序设计语言，是第一个广泛使用的高级编程语言。在企业管理中，数值计算并不复杂，但数据处理信息量却很大。为专门解决经企管理问题，于 1959 年，由美国的一些计算机用户组织设计了专用于商务处理的计算机语言 COBOL，并于 1961 年美国数据系统语言协会公布。经不断修改、丰富完善和标准化，目前 COBOL 已发展为多种版本。

51CTO：这几年的工作经历中，您觉得在技术上提高的最大困难是什么？

李先生：和客户的沟通，干对日外包这行，基本上是客户让你干你就怎么干，两方交流主要是靠邮件，不能打电话，很多问题，尤其是语言上的理解不一样，造成沟通障碍。

51CTO：对于刚接触对日外包的新生，您有怎样的建议？

李先生：脚踏实地，细心，认真，灵活

已经有不少对日外包程序员提到了要脚踏实地的工作，坚持做好一门工作看来是很多程序员的准则了。

51CTO：请谈谈在日本的生活？

李先生：经常加班，而且日本人是有加班费，所以日本人喜欢加班。

在看过日本程序员的生活后，发现日本白领下班后自动加班的愿望很强烈。在这种氛围下，任何按时下班的人都会被看做异类，也被看做是不积极上进的代表。不过高效率的工作，把事情在 8 小时内解决不好么？

外包程序员晋升项目经理

编者按

对日软件外包相信大家并不陌生，但是这个行业的程序员究竟是怎样的一种状态？我们想通过他们自己的话来让大家有更深刻的了解。

刘先生曾在一家国内著名对日外包软件开发公司担任项目经理，现在日本东京一家著名软件开发公司工作，对日外包被日本老板挖走的成功转型案例（地震后对继续留在日本还是回国十分纠结。

能从外包行业直接转到甲方，确实需要很深的技术功底以及良好的日语基础。其实很多对日外包程序员技术扎实，但是在与日方沟通上还是存在问题。日语不会，对日本文化了解不深造成了隔阂。

51CTO：您从事对日外包已经几年？当初是怎么进入这一行的？

刘先生：已经从事对日外包10年，当时是工作需要。

51CTO：你平时使用的开发技术？目前您正在做哪方面开发工作？

刘先生：Java，Visual Basic。日本公官厅。

51CTO：在这几年的工作经历中，您觉得在技术上提高的最大困难是什么？

刘先生：两国文化差异

51CTO：您认为语言和文化习惯是不是阻碍对日外包工程师晋升的障碍？

刘先生：40%-50%吧。

51CTO：对于刚接触对日外包的新生，您有怎样的建议？

刘先生：技术，日语都要过关。

51CTO：对日外包与国内的项目有什么区别？

刘先生：认真谨慎，不要图于快，质量是最重要。日本关于信息安全的要求特别高，比如你在日本干活，和其他项目组的人不能说关于本项目的内容，公司的文件不能带出去，不让上外网，而且上班不能用笔记本和存储设备，这一点相当严！

51CTO：谈谈在日本的生活？

刘先生：软件，做日本软件很辛苦，几乎每天都要加班，日本人做事情按部就班，认真谨慎，能互相建立起信任感之后，工作变得简单而且轻松，但是这种信任感需要有1-2个项目或者是1年的时间才可以建立起来。

毕十年之功，打好基础是一个程序员走向项目经理的必经之路。其实日本人不是太谨慎，而是目前中国国内软件开发行业浮躁之气太甚。软件BUG多，难以重构和二次利用都是浮躁的体现。上面刘先生讲到要有1到2个好项目才能建立良好的关系，也是值得大家去注意的。

能做到项目经理是每个程序员的目标，也算是个人职业生涯的一个高峰。但为什么很多程序员做不到项目经理，这恐怕还是大家不会与人沟通，不了解管理和队伍建设。

只顾自己闷头开发是大忌。

给对日外包程序员的小建议

认据调查中国有 60% 的软件和服务外包业务来自日本，所以中日两个在软件和服务的外包领域关系十分紧密。目前在中国，大连是承接日本外包业务最多的城市之一，而且还有不少城市在该方向上努力向其看齐，所以对日外包程序员的基本量还是非常大的。那么，对日外包程序员该如何发展？不同层次的程序员又该如何突破发展瓶颈，如何提升技术和业务能力呢？

对日外包其实是软件外包领域中较为特殊的，特殊的原因则是文化的不同。日本人在工作方面喜欢书面化、流程化，所以我们会发现，日本的外包项目中通常包含有十分明确的完全书面化的需求、流程定义、开发计划、测试计划甚至框架定义等式样书。这些严格标准话的式样书对软件的质量保证和工期成本控制都是很有益处的，也是值得国内很多软件公司借鉴的，但也存在一些负面影响——比如，程序员的职业发展。

51CTO 特约评论员

衣明志，蝉联五届微软 MVP，51CTO 特约评论员，烟台 .NET 俱乐部主席，现主要从事 .NET 平台下的 Web 应用开发、解决方案、构架设计及技术培训等工作。

笔者在刚毕业时也曾在日资软件公司工作过一段时间。那时所用的操作系统、开发工具、式样书统统都是日文的，而各个项目小组的 Leader 基本也都是日本人，所以语言会是一个重要的问题，要么你要懂日文要么让日本人懂中文——日本人的英文普遍较差。当时发现公司的日本程序员 90% 都不是计算机专业，有会计专业、财务专业、工商管理专业的，甚至还有美术和音乐专业的。开始时对这些日本程序员的主要专业方向很是困扰：一个不懂计算机技术的程序员能够做好一个软件系统吗？但是当我们开始正式开发时，我们发现各小组的人员接触到都是定义及其明确而严格的式样书甚至函数定义，需要程序员做的大部分就是 coding，而行业软件要求的技能甚至比技术技能更高。所以基本不用关心你 code 用在哪个地方等等问题，也不用考虑按钮的效果和位置是否合理，因为样式书都有定义，事无巨细。所以对于刚进入或准备进入对日外包企业“新兵”们，最好先把日语好好学习。

给对日外包程序员建议 II

作者 / 衣明志

通常培训学校和相关企业都有这方面的专业培训，外加自己努力，尽可能让自己的日语到二级以上，那么你将来在对日外包这个领域才能相对吃的开。但也不要忘记，你是做技术工作的，不是翻译，所以技术能力仍然是你将来的重要能力方向，至少技术不错还可跳槽到其他类型的软件公司嘛。

在对日外包领域工作几年后，很多开发人员会出现一些问题：过分重视日语，忽略技术能力；各种技术样样通，样样不精；过分专注 Coding，忽略项目整体观；过分专注业务流程，忽略代码的执行效率等等。

所以笔者在这里给已经在做对日外包工作的开发人员一些建议：

第一，日语在对日外包领域中是必需的基本技能之一，但不是唯一，否则公司招聘日语专业或者日本留学生即可。技术技能的提升绝对不要忽略。

第二，利用尽可能多的空余时间了解技术动态、提升技术技能，对某项语言或技术做重点学习和研究，努力成为某技术的专家，并了解或熟悉其他技术。

第三，　由于对日外包项目中，大部分人仅仅接触项目中的一些底层工作，所以比较少能看到整体项目的一些情报。为了职业发展，建议多找一些机会了解整体项目的情况，并尽可能争取一些独立完成小项目的机会，提升项目开发技能，而不仅仅熟练 Coding。

第四，对日外包项目中，很多对于流程严格规定，但对性能的要求则不甚严格，这大概跟前面提到的日本非计算机人员开发软件的普遍状况有关。但我们作为计算机专业人士，应该对自己提出更高的要求，在达到基本要求的同时，考虑是否可以将代码执行效率做进一步提高？原来运行 1 小时才能出结果，能否提升到 1 秒钟？考虑现在的式样书中的流程或算法定义是否存在改进空间，并在空余时间考虑改进方案的实现，或提交上司知晓。

第五，对单语种的软件开发，时间长了，会丧失一些国际观。建议有兴趣的人，多关注一下系统国际化方面知识和技能。比如：各种文件和语言编码对系统的影响；各国的货币、日期、时间的表示形式；时区问题；支持多语言包的系统设计……

第六，理性对待加班问题。日本人大都具有强烈的自动加班愿望，笔者认为主要是两个原因：正式工几乎不允许做其他兼职工作，而在公司内加班有加班费，所以从经济压力上来说日本人加班具有原动力；下班后，日本的上司通常会习惯性地查看员工工作情况，加班的员工被习惯性地认为是积极上进的表现，所以日本人认为加班会提升自己在上司心里的位置，有利于升迁。所以，一些公司和场合日本人加班属于习惯性的，两种加班原动力会促使日本人降低正常工作时间的工作效率，以便加班时确实有工作可做。

给对日外包程序员建议 III

作者 / 衣明志

那么我们要搞清楚具体状况，合理地处理加班问题，不要顾此失彼或人云亦云。

如果有幸，你成为了项目经理或者准备成为项目经理，那么建议重点提升一下两个方面：沟通能力和组织管理能力。也许有人会说，项目的设计规划方面也要提升！

是的没错，但是对日外包中，项目经理拿到的项目通常也是设计规划好的，所以你很难在该方面有重大突破。对项目经理而言沟通非常重要，所以日语最好达到一级，并最好真的跟一些日本人长期交流过，熟悉他们的文化和习惯，并与客户建立良好的信任基础。

沟通中的语言技巧和肢体表现也应该多学习一下，开发人员很多人存在沟通障碍，缺乏语言技巧。所以，笔者建议在项目经理职业道路上的技术人员，平常多锻炼语言表达能力，学习演讲技巧甚至销售技巧，以便能够大幅提升您在客户、上司和下属心中的位置，起到上下信息畅通的作用。沟通能力的提升对于组织管理也很有益处，通过良好的沟通可以化解各方面的矛盾和问题，提升团队凝聚力和工作效率，对项目控制起到很好的效果。

最近日本接连发生地震、海啸、核危机等重大灾害，使得日本的实体经济受到一些影响，尤其是在日本东北地区灾害比较严重，所以一定程度上会造成日本外包项目的订单减少。但由于重灾区主要是日本的经济落后地区，所以整体外包市场影响不太严重。而在日本本土工作的人员，近期因灾难因素回国的并不占多数。但最近有消息称日本的部分火山活动频繁，需提高警惕，国内的对日外包从业者应该密切关注相关事件的发展，做好预防和应对措施。另外积极考虑欧美外包市场，扩大行业领域和区域分布，实现订单来源多元化，分散业务风险，同时关注因灾难带来的新机遇，比如：云计算、云存储。灾后重建，日方企业会考虑在一水之隔的中国大陆建立备灾中心，以降低未来灾难对企业的负面影响，提升企业健壮性和市场竞争力。所以无论是对日外包的企业还是相关从业者都应该关注云计算等新兴领域的市场，积极应对，争取抢到第一桶金。皆さん，顽张ってください！

开发小 TIP

我们在 HTML 5 中是否使用双引号

这有点让人纠结，HTML 5 并不是 XTHML，你可以省去标签中的双引号。相信大多数同志习惯了加上双引号，因为这让代码看起来会更标准。不过，这可以根据你的个人喜好来确定是到底要不要双引号。

大家可以参考下面的 HTML 5 代码

<h6 id="someid" class="myclass"> start the reactor. </h6> 其实代码可以写成：

<h6 id=someid class=myclass> start the reactor. </h6>

■ 编者按

本文是 Common Lisp 专家 Peter Seibel 对 Google 公司首席 Java 架构师 Joshua Bloch 的访谈，谈到他所遇到的最糟糕的 Bug 以及 Java 的命运。

Google首席架构师谈Java的命运

Joshua Bloch，现任Google公司首席Java架构师。之前他在Sun公司工作，领导并实现了Java 2中的Java Collection Framework，还参与了Java 5发行版中几项语言附加特性的设计。Bloch在卡内基梅隆大学获得博士学位，期间他参与设计了Camelot分布交易处理系统，后来演变为Transarc公司的产品Encina，而他则是Transarc的资深系统设计师。他编写的Effective Java一书获得了2001年Jolt大奖，他还与人合著了《Java解惑》和《Java并发编程实践》。

最糟糕的 Bug

Seibel：我们聊聊调试吧。你遇到的最糟糕的 Bug 是什么？

Bloch：提起 Bug 我立马就想到了一个，这个 Bug 很严重，而且很搞笑。那是 90 年代初，我在匹兹堡的 Transarc 公司工作时。我在很紧的工期下提交了一个事务共享内存的实现。我在限期内完成了设计和实现，甚至还在过程中做出了几个可重用的组件。但是这么匆忙地写了很多新代码，我还是挺担心的。

为了测试这些代码，我写了一个叫做“乱撞”的很长的程序出来。它运行了大量的事务，每个事务又包含了嵌套的事务，嵌套到可以嵌套的最大深度。每个嵌套事务都可能会加锁，以递增的顺序读取共享数组里面的几个元素，对每个元素都加入点东西，保持数组中所有元素的和为 0，还是不变量。这些事务要么提交，要么取消，如 90% 的提交，10% 的取消，其他比例也可以。多个线程同步运行于这些事务之上，长时间地访问数组。因为我测试的是一个共享内存机制，所以我同时运行多个有多个线程的“乱撞”程序，每个有自己的进程。

在一般的并发级别下，“乱撞”轻松过关。但是当我真正调高并发级别时，我发现“乱撞，偶尔，仅仅是偶尔，无法通过一致性检查。我不知道这是怎么搞的。这只能是我的错，因为新代码都是我一个人写的。我花了大约一个星期，痛苦地为每个组件写了彻底的单元测试，所有的单元测试都通过了。然后我为每个内部数据结构写了详细的一致性检查，这样我就可以在每次变化后调用这些一致性检查，直到测试失败为止。最后，我终于发现一个底层的一致性检查失败了，这个问题无法重现，但是某种程度上可以帮助我分析问题出在哪里。

Google 首席架构师谈 Java 的命运 II

最后，我得出了确实的结论——我的锁根本不工作。两个事务锁定、读写同一个值的时候，产生了并发的读—修改—写回操作，而后一次写入毁掉了第一次的写入。

我编写了自己的锁管理器，所以我怀疑是它出了问题。但是锁管理器轻松地通过了测试。最后，我觉得问题不在锁管理器，而是它依赖的互斥体的实现！那时候操作系统还不支持多线程，我们需要写自己的多线程包。原来负责互斥体代码的工程师，不小心把我们的 Solaris 的线程实现中的 lock 和 try-lock 的汇编代码的标签弄混了。所以，每次你以为你在调用 lock 的时候，其实调用的是 try-lock，反之亦然。也就是说当真的有争用发生的时候——在当年其实是很罕见的——第二个线程直接就进入了第一个线程的临界区，因为第一个线程也没有锁住。搞笑的是，这也就是说，整个公司几个星期都在运行没有互斥体的程序，而且谁都不知道。

Knuth 有句关于测试的名言，Bentley 和 Mcllroy 的精彩论文“Engineering a Sort Function”中曾经引用过，大概意思是说，做测试时，要不惮以最大的恶意来推测所要测试的代码的错误。做这些测试的时候，我就是这么做的。但是这样会把所有东西纠结在一起，更难找到 Bug。首先，并发的时候很难这么做，往往完全无法复现场景。其次，到最后可能会发现你的核心假设是错的。喜欢喊“耶，这语言出错了”或者“系统出问题了”是新手干的事儿。但是在这里，我依靠的基石——互斥体，确实出问题了。

Seibel：也就是说 Bug 不在你的代码中，但是同时你只能对你的代码进行彻底的单元测试，因为你没有别的办法，只能去检查自己的代码。你觉得这些测试是不是可以，或者说应该让互斥体代码的作者来写，这样你就不用浪费一个星期，节省了一半的测试量，而且也可以找到这个 Bug。

Bloch：给互斥体代码加一个好的自动化单元测试肯定可以避免让我遭受那些痛苦，不过注意那可是 90 年代初。我想都没想过要抱怨那个工程师没写个好的单元测试。即使是今天，为并发工具写单元测试还是一种艺术形式。

Java 的命运

当你改进一个成熟语言的时候，你必须更加仔细地考虑能力和复杂度之间的平衡。

Seibel：既然你说到这里，Java 是否逃脱了灭亡的命运了？它变复杂的速度是不是比变好的速度更快呢？

Bloch：这个问题不好回答。具体说来，Java5 加入了比我们设想的更多的复杂度。将泛型特别是通配符加到语言中到底有多复杂我也说不好。我得为有功劳的人说句话，GrahamHamilton 真是了不起，那时候他就想明白了一切，而我不明白。

有趣的是，他抗争多年，希望阻止泛型进入 Java 语言中。但是在泛型被成功地阻挡在 Java 外的这些年里变体的概念，也就是通配符的隐含意义流行了起来。如果它们来得更早，没有变体，也许我们现在可以有一个更简单的、更容易跟踪的语言。

引入通配符有实际的好处。子类化和泛型

Google 首席架构师谈 Java 的命运 III

之间根本就是阻抗不匹配的，通配符尽力在弥合这种不匹配。但是这么做又显著地增加了复杂度。有些人认为在声明空间，而不是用户空间，变体是更好的解决方案，但我不太相信这一点。

这仍旧悬而未决，因为它们都还没经过在真实世界里海量的程序员们的测试呢。一些语言经常只在小范围内获得成功，人们会说："噢，这些语言很不错，只是可惜没有成为世界范围成功的语言。"但是这往往是有原因的。希望使用 Scala 或者 C#4.0，这样的声明空间变体的语言可以彻底解决这一疑问。

Seibel：那么是什么推动 Java 引入泛型呢？

Bloch：没看起来那么精彩啦，我们的新闻报道是可信的。我的思维模式是，"嗨，集合多半都应该是同质的——一组字符串，一个从字符串到数字的映射，等等。而现在默认情况下集合是异质的：它们都是对象的集合，取出时都需要类型转换，这简直是胡闹。"如果我可以告诉系统，这是一个从字符串到数字的映射，它会帮我做类型转换，而且会在编译期间帮我盯着，防止我做错什么，那不是挺好的吗？它可以抓到更多的错误——它可以包含高层的类型信息，看起来是件好事儿。

我认为泛型和其他加入到 Java5 的语言特性一样，我们只是让语言去做以前我们要手工去做的事情而已。某些情况下我坚信：foreach 就是好。它所做的就是对你隐藏遍历器和索引变量带来的复杂性。代码更短，概念也不复杂。从某种意义上说，它的概念更简单，因为我们为数组和其他的集合创建了这种伪多态机制，你可以遍历一个 ArrayList 或者一个数组，而无需关心你遍历的是什么类型。

这种思想不能适用于泛型的主要原因是，它是对已经很复杂的类型系统的大扩展。类型系统是很微妙的，修改它们可能对语言带来深远的、难以预期的影响。

我认为得到的教训是，当你改进一个成熟语言的时候，你必须更加仔细地考虑能力和复杂度之间的平衡。而且，实际上，复杂度跟语言的功能数量间至少是平方级关系。为一门老语言加上了一个新的功能，通常就意味着为它加入了一大堆复杂度。当一种语言已经达到或接近程序员理解能力的极限时，那么你加入任何复杂性进来都会加剧理解的难度。

语言更复杂后就会消失吗？不会。我认为 C++ 早已超越了它的复杂度极限，但还是有很多人用它编程。可这实际上是逼人们只使用其中一个子集。所以我认识的每个用 C++ 的公司都说："对，我们用 C++，但是用的是多继承，不用操作符重载。"有很多功能你完全不用，因为使用它们会造成代码太复杂。即使不得不用那些功能，我认为也实在没什么好处。那样的话，程序员就读不懂别人的代码，也就不存在"程序员的可移植性"了。

Seibel：如果去掉泛型，现在 Java 会变得更好用吗？

Bloch：我不知道。我还是喜欢泛型。泛型能帮我找到代码中的 Bug。泛型可以让编译器强

Google 首席架构师谈 Java 的命运 IV

制做一些限制，之前这些限制我只能放在注释中。另一方面来说，当我看到那些疯狂的参数类型相关的错误信息，当我看到像 classEnum> 这样的泛型类型声明时，我就会想，显然泛型的设计还没成熟到可以放到 Java 中的水平。

我们总是太乐观，然后搬起石头砸自己的脚。所以我们说："耶！我们当然可以把泛型放到 Java 中。在 CLU 的时候我们就知道泛型了。这技术 25 年前就有了。"最近我听到关于闭包的类似言论，不过那是 50 年前的技术了。"噢，闭包很简单，不会给语言加入任何新的复杂性。"

嗯，没错。但是我觉得我们从泛型这件事儿得到了教训。在你懂得这个改动会对概念层面带来什么影响之前，在你可以确保软件行业从业人员可以高效使用新特性，而且这一新特性会让他们活得更好之前，你不应该给语言加入这一特性。

如果早知道程序员们对泛型是这个反应，我们肯定不会把它加到 Java 里。这是不是说我们就完全不会搞泛型？不，我不这么认为。我认为泛型确实很好。主要是因为大多数集合是同质的，而不是异质的，同质的集合处理起来是比较方便的。多数情况下类型转换都不合适。转换可能会失败，而且让你的程序不再优雅。我想你知道这是什么集合，它应该自动符合你的这些需求。但是，是不是这就意味着你应该承受我们现在承受的这种复杂度？不，我想我们只是没有处理好泛型。

Seibel：关于泛型有来自于用户的压力吗？有人抱怨泛型的缺点干扰他们写程序了吗？

Bloch：有没有工程师大骂泛型的缺点？不，没有，他们没有抱怨过。如果因为泛型简洁就把它们加进来，那我会内疚的。因为当时我们以为这么做是对的。

有人说，很多工程都是扯淡。有人要求我们加入 foreach 吗？没有。他们没有要求我加入。但是我就知道这是应该做的。我对了——每个人都喜欢它。但是我觉得我们行业内的一大问题就是，在工程领域，做一个东西，仅仅因为它简洁，仅仅因为它是一个好的工程项目，等等。如果你不能解决真实用户——在这里就是 Java 程序员——的真实问题，那么你就不应该加入新的特性。

James Gosling 曾做过一个非常了不起的演讲——"Java 的感觉"。他说，给 Java 加入任何东西之前，都需要三个真实的用户。不应该因为一个东西简洁就把它放进来。

但是人们就是想把什么东西都放进去。工程师是做什么的？他们就是写代码的。而当他们写一个库，或者一个语言的时候，他们就是想放各种东西进去。你需要他人的参与，需要指导的声音，需要这些东西来帮助你完成产品，帮你在放与不放之间做出最好的权衡。因为你可以放进去的东西总比你应该放进去的东西多。那么是不是说所有的这些东西都不好呢？那也不是。只是你需要做出决定，某些东西是不应该放进去的。

思考 Java 带来的编程经验

Seibel：思考 Java 的设计并实现它，是否让你学到了什么跟编程有关系的东西？

Bloch：我学到的东西太多了。比如我知道了

Google 首席架构师谈 Java 的命运

即使是想把一个很小的程序写对也是非常难的。我把这个想法发表在了博客里，题目是“几乎所有的二分搜索和归并排序都是错的”。认为自己程序是对的就是在愚弄自己。程序里有大量没解决的 Bug，当然是不对的。多数情况下，程序里的 Bug 都不少，它们只能免费完成任务。

我知道，既然写正确的程序那么难，我们就应该尽力去帮助大家。所以能减少 Bug 的所有东西都是好的。这就是我是静态类型和静态分析的信徒的原因，任何可以减少某个特定类别 Bug 的东西都是非常好的，任何可以让程序员的工作更轻松的东西都是好的。

我更加确信有好的 API 文档是很重要的。人们很少提及 Javadoc 对这个平台的成功所起的作用。好的 API 文档永远都是 Java 文化的一部分，也许是因为 Javadoc 从一开始就存在吧（译者注：所以人们低估了 Javadoc 的作用）。

我一直信奉“简单就是美”这句话，现在更是如此。我不断看到更复杂的东西最终被证实是有害的，只是有的时间长点儿，有的时间短点儿。我设计的时候，会仔细看着我的“复杂度计”，一旦复杂度要到红线了，就需要重新设计了。

偶尔我会遇到不相信这些的人们，他们会说：“Josh 你太傻了，你怎么就是不明白；这才是应该做的，可惜你就是搞不懂。”我就是不信这些。我觉得事情一旦复杂起来，那么一定有什么地方错了，也许到了寻找更简单的方法的时候了。

Tony Hoare 的图灵奖获奖感言中有一句充满了大智慧的话，讲的是设计一个系统的两种方式：“一种是尽量简单，这样显然不会有什么问题；另外一种是，尽量复杂，这样没什么问题会很显然。”

后面的内容同样饱含智慧，但是知道的人不多：“第一种方法其实更难。它需要从复杂的自然现象发现简单物理规律的那种技能、投入、洞察力，甚至是那种灵感，同时还需要你能接受你的目标受限于物理、逻辑和科技的约束，以及在目标间有冲突的时候可以妥协。委员会无法做到这些，除非已经完全来不及了。”

我不知道将出现什么。但是我认为如果改变我的主要语言是对的，那么我就会这么做。我想尽力保持开放的心态。我想尝试更多的语言。我最近没时间做，但是以后还是会做的。

Seibel：现在很多人在讨论我们写程序的时候，如何能把未来的多核 CPU 的优势利用起来。Java 显然是第一个内建多线程机制的主流语言。你觉得 Java 的逻辑在多核的世界是否仍然可用？

Bloch：我想说得更深入一些。我认为 Java 是现有语言中最好的。但有趣的是，现在很流行谈 Java 是否即将死去。我觉得这基本上是扯淡。我认为现在最好的多线程构件就在 Java 里。我认为 Java 将迎来复兴。我不是说它是未来 20 年内最先进的，也不是说它是处理多核的最好方式。但是我认为从现有的东西来看，我们是足以傲视同侪的。

http://developer.51cto.com/art/201103/248752.htm

（本文摘自《程序员》杂志）

.NET程序员应该放弃VB.NET?

VB.NET 让很多 .NET 程序员又爱又恨,放弃 VB.NET 似乎可以让程序员们的生活更加轻松。但是你真的愿意放弃它吗?

我是否该放弃 VB.NET 呢? 这个问题一次次的出现在我的脑海里,而且这种想法越来越强烈。放弃 VB.NET 至少能让我的生活变得轻松些。如果你是个 C# 程序员,那拷贝粘贴代码会很容易,因为可以找到的例子代码如此的多。C# 社区越来越大。甚至微软也不鼓励再使用 VB.NET。如果你去一些网站,如 Techdays,你根本找不到 VB.NET。在那个社区里,如果你告诉他们你是一个 VB.NET 程序员,你会受到他们的挖苦和嘲弄。你会被瞧不起,连那些菜鸟们也瞧不起你。.NET 社区看起来有些粗鲁、不友好——不论你做了什么。我是在周末开发我的小 powershell 时发现这个现象的。

所以,选择 VB.NET、选择 Winforms 是不是一个错误的选择? 这个问题出现在我的脑子里。我估计 VB6 社区里的人也会有同样的疑问。包括 Delphi 社区。很显然,就连 WPF 社区都已经感觉到人们的宠爱正在消退。你是否注意到这些都是微软阵营的。微软习惯于干那些创造一个事物,然后为了下一个伟大的事情抛弃这个,甚至不做任何努力来帮助人们把旧代码迁移到新架构上。

你知道想在网上找一个拥有不错的博客的 VB.NET 开发人员有多么的困难吗? 我估计他们大部分都把博客贡献给了 C#,去获取更多的 C# 经验,或他们什么都没写。

我曾试图说服 Scott Hanselman 在他为 2011 Belgian Techdays 做的大纲里加几句 VB.NET 的话。就好象是一场攻占某个高地的战斗。参考一下这个投票结果,我想现在他更没有兴趣了。

我不责怪他们,他们都是要去赚钱的生意人,他们必须做他们自己认为该做的事情。

看看上面的所有这些原因,我是否该下决定做改变呢? 不,目前不会。我会坚持使用 VB.NET 和 winforms,直到它们还能用、我还能坚持。为一个新的技术的产生而重写一个程序并不是一个明智的行动,不管从短期或长期看,它都会让你受损失。你最好还是在你现有的应用上添加新功能。

当这种事情出现时,我第一想到的是它是否会给你现在的处境增加有利的价值。不要为了改变而改变,要为更好而改变。请不要忘记历史,即使那些最优秀、最聪明的人也犯过这样的错误。

我感觉随着年纪的变老,一次次的,我开始变得沉着冷静了。

http://developer.51cto.com/art/201103/248499.htm

HTML 5革新：结构之美

作者 /iefans

HTML 5 如同一场革命，正在 Web2.0 后时代轰轰烈烈的进行着。

HTML 5 是什么，无须我在这里赘述了。对于 HTML 5 的革新，按我的理解，可以总结为语义明确的标签体系、化繁为简的富媒体支持、神奇的本地数据存储技术、不需要插件的富动画(canvas)、强大的 API 支持。总之，HTML 5 让人机交互，人网交互变得更加舒适，贴合用户。以往对富媒体应用与本存储的支持乏力也不再是浏览器的切肤之痛。将 Web 从内容平台推向标准化的应用平台，并一统各在平台阵营的标准，才是 HTML 5 革命的初衷。

本文，我就抛砖引玉，阐述 HTML 5 的革新之一：语义更明确简洁的结构。

从”头”说起

一个标准的 XHTML 头部代码应该是这样：

```
<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML 1.0 Transitional//EN” “http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd”>
<html xmlns=” http://www.w3.org/1999/xhtml” >
<head>
<meta http-equiv=” Content-Type” content=” text/html; charset=gb2312″ />
</head>
```

你能记住吗？你会去死记硬背吗？当然不会！我们只需要机械的复制粘贴即可。

再看看一个标准的 HTML 5 头部是如何的：

```
<!doctype html>
<meta charset=gb2312>
```

孰繁孰简，就不用我说了。是的，HTML 5 的头部可以如此简单，可以轻易的记住！并且，可以忽略大小写，引号以及最后一个尖括号前的反斜线。

为什么可以如此松散？其实，如果把 XHTML 当成 text/html 发送，浏览器一样可以很好的解析，浏览器并不在乎代码的语法。所以，HTML 5 是形而上的，它可能会破坏原有的一些标准，但仍可在浏览器中很好的表现。

当然，为了团队协助与后续维护的方便，我们还是应该统一一种你喜欢的风格的写法，比如：

```
<!doctype html>
<html><head>
<meta charset=” gb2312″ />
…
</head><body>
…
</body></html>
```

另外，HTML 5 虽然目前并不为所有浏览器所支持，但这个能省去 100 多字节 (对于日 PV 百万级以上的站点，能省下不少的流量哦) 的头部已可以完美的兼容了。如果你对浏览器解析模式有研究的话，你应该知道，页面在没有定义

HTML 5 革新：结构之美 II

doctype 的情况下会触发怪异模式，而只要定义了 <!doctype html> 浏览器就可以在标准模式下解析页面，而不需要指定某个类型的 DTD。

新的语义化标签体系

语义化编码是一个合格前端 Developer 必备的技能，但随着网页的日渐丰富化，仅仅用原有的 xhtml 标签去语义化显然已经力不从心。上帝说：”要有光！”便有了光。于是，HTML 5 提供了一系列新的标签及相应属性，以反应现代网站典型语义。实践出真理。还是写一个例子吧：

```
<div id=" header" >
<div class=" hgroup" >
<h1> 网站标题 </h1>
<h1> 网站副标题 </h1>
</div>
<div id=" nav" ><ul>
<li>HTML 5</li>
<li>CSS</li>
<li>JavaScript</li>
</ul></div></div>
<!–//header end–>
<div id=" left" >
<div class=" article" >
<p> 这是一篇讲述 HTML 5 新结构标签的文章。</p></div>
<div class=" article" >
<p> 这还是一篇讲述 HTML 5 新结构标签的文章。</p></div></div>
<!–//left end–>
<div id=" aside" >
<h1> 作者简介 </h1>
<p>Mr.Think，专注 Web 前端技术的凡夫俗子。</p></div><!–//side end–>
<div id=" footer" >
页面的底部 </div>
<!–//footer end–>
```

上面是一个简单的博客页面部分 HTML，由头部、文章展示区、右侧栏、底部组成。编码整洁，也符合 XHTML 的语义化，即便是在 HTML 5 中也可以很好的表现。但是对浏览器来说，这就是一段没有区分开权重的代码，而不是一个让机器也能读懂语义的标签来定义相应的区块。比如，标准浏览器（比如 Firefox、Chrome 甚至新版的 IE9）都有一个快捷键可以带引客户直接跳转到页面的导航，但问题是所有的区块都是用 DIV 定义，并且 DIV 的 ID 值是同开发者定的，所以，浏览器并不知道哪个应该是导航链接所在区块。HTML 5 新标签的出现，正好弥补了这一缺憾。那么，上面的代码，换成 HTML 5 就可以这样写：

```
<header><hgroup>
<h1> 网站标题 </h1>
<h1> 网站副标题 </h1>
</hgroup><nav><ul>
<li>HTML 5</li>
<li>CSS</li>
<li>JavaScript</li></ul></nav></header>
<div id=" left" >
<article>
<p> 这是一篇讲述 HTML 5 新结构标签的文章。</p>
```

HTML 5 革新：结构之美 III

```
</article><article>
<p> 这还是一篇讲述 HTML 5 新结构标签的文章。</p></article>
</div><aside>
<h1> 作者简介 </h1>
<p>Mr.Think，专注 Web 前端技术的凡夫俗子。</p></aside>
<footer> 网页底部 </footer>
```

原来，HTML 的页面结构可以如此之美，不用注释也能一目了然。对于浏览器，找到对应的区块也不再会茫然无措。

如何用 HTML 5 新标签结构化元素

通过上面的示例，我们了解到 HTML 5 的新标签对结构化的革新，但切换到实际使用中，该如何恰当的使用它们呢？我想这也是很多 HTML 5 学习者想问一个问题。如同 XHTML 语义化一样，HTML 5 语义化标签的使用也应该遵循：每个标签都有它特定的意义，而语义化，就是让我们在适当的位置用适当的标签，以更好的让人和机器（机器可理解为浏览器可理解为搜索引擎）都一目了然。

比如 header 标签一般是页面的第一个区块元素（header 标签也可用于类型的头部元素中，比如文章区块的标题），包含的了页面的主题信息；nav 标签一般用于包裹导航信息；footer 一般用来包裹页面底部信息；等等。

下面是我参考 HTML 5 手册列出的结构类常用新标签的语义解释及使用指引：

<header> 标签

手册释义：定义 section 或 document 的页眉。

使用指引：一般用来包含页面头部，也可用于其他区域头部，比如 article 头部：

```
<header><hgroup>
<h1> 网站标题 </h1>
<h1> 网站副标题 </h1>
</hgroup></header>
```

<hgroup> 标签

手册释义：用于对网页或区段（section）的标题进行组合。

使用指引：用于标题类的组合，比如文章的标题与副标题：

```
<hgroup>
<h1> 这是一篇介绍 HTML 5 结构标签的文章 </h1>
<h2>HTML 5 的革新 </h2>
</hgroup>
```

<nav> 标签

手册释义：定义导航链接的部分。

使用指引：用于定义页面的导航部分：

```
<nav><ul>
<li>HTML 5</li>
<li>CSS</li>
<li>JavaScript</li>
</ul></nav>
```

<aside> 标签

定义 article 以外的内容。aside 的内容应该与 article 的内容相关。

使用指引：用于成节的内容，会在文档流中开始一个新的节，一般用于与文章内容相关的边栏：

HTML 5 革新：结构之美 IV

```
<aside>
<h1> 作者简介 </h1>
<p>Mr.Think，专注 Web 前端技术的凡夫俗子。</p>
</aside>
```

<section> 标签

手册释义：定义文档中的节（section）。比如章节、页眉、页脚或文档中的其他部分。

使用指引：用于成节的内容，会在文档流中开始一个新的节：

```
<section>
<h1>section 是什么？ </h1>
<h2> 一个新的章节 </h2>
<article>
<h2> 关于 section</h1>
<p>section 的介绍 </p>
…
</article></section>
```

<footer> 标签

手册释义：定义 section 或 document 的页脚。典型地，它会包含创作者的姓名、文档的创作日期以及 / 或者联系信息。

使用指引：一般用来包裹整个页面通用底部，也可用于其他区域底部，比如 article 底部：

```
<footer>COPYRIGHT@ieFans.Net</footer>
```

<article> 标签

手册释义：定义外部的内容。比如来自一个外部的新闻提供者的一篇新的文章，或者来自 blog 的文本，或者是来自论坛的文本。亦或是来自其他外部源内容。

使用指引：顾名思义，一般用于文章区块：

```
<article><header><hgroup>
<h1> 这是一篇介绍 HTML 5 结构标签的文章 </h1>
<h2>HTML 5 的革新 </h2></hgroup>
<time datetime=”2011-03-20 ”>2011.03.20</time></header>
<p> 文章内容详情 </p>
</article>
```

<figure> 标签

手册释义：用于对元素进行组合。

使用指引：多用于图片与图片描述组合：

```
<figure><img src=” img.gif” alt=” figure 标签” title=” figure 标签” />
<figcaption> 这儿是图片的描述信息 </figcaption></figure>
```

<menu> 标签

手册释义：定义菜单列表。当希望列出表单控件时使用该标签。

使用指引：使用于菜单类区块，用来定义菜单列表或菜单选项：

```
<menu><li>HTML 5</li>
<li>CSS</li>
<li>JavaScript</li>
</menu>
```

HTML 5 的其他新标签，就不此一一解释了，请自行查询一下手册。其实，这些东西，如同 XHTML 的 div、h1、inpu 等标签一样，只要平时多加实践，运用自如也是轻而易举的。

HTML 5 革新：结构之美 V

关于兼容性

如果你是一个喜欢研究关注前端的人，你应该知道淘宝的页面结构中已大量用到了 HTML 5 新标签。所以，我想说的是只要敢于尝试，兼容性不是问题，兼容的方法，网上有很多（本文是讲结构的）。

后话

任何一门新技术，都需要一个适应的过程。如果你准备好了做一名优秀的 Web 前端开发人员，那你就得不断的尝试并接受最新的前端技术。

孙文曾说，欲经文明这幸福，不得不经文明之痛苦。人类的革命如此，HTML 5 的革命亦是如此。IE 的日渐没落，让各大浏览器厂商以一次进入了战国时代，群雄逐鹿。而对于开发者，我们只奢求各大浏览器厂商尽可能的遵循同一个标准，而不是群雄逐鹿后的四分五裂。因为，高效完美的呈现给各类用户同样的应用才是我们的终极目标。

如此，本文从页面的 doctype 说起，到用 HTML 5 新标签搭建语义化更明确的页面的结构，再解释了一番与页面结构相关的新标签。相信大家对 HTML 5 的结构性新标签有了一个新的认知，如果你有兴趣，那就打开你的 IDE，写上一段由 HTML 5 新标签搭建的代码，然后用 CSS 去描绘你的宏伟蓝图吧！

延伸《HTML 5 标准学习入门之文档结构》

说起 HTML 的结构，很多人都能说得头头是道，一般来说答案可能是这样的：

一个 DOCTYPE，一个 html，里面有 head 和 body 元素。

这当然不能说是不正确的，但是如果问到一个最小的 HTML 源文件必须有哪一些东西的话，恐怕很少有人能正确地做出回答。

先来回答一下这个问题，一个最简的 HTML5 源码文件需要的内容如下：

```
<!DOCTYPE html>
```

是的，就这样，一个字符不多，一个字符不少，除了大小写可任意变化外，其他的任何内容都是不能变动的。

那么究竟是怎么样的规则，导致一个最简的源码文件必须有 doctype 声明呢？根据标准，一个 HTML 文档有如下内容组成（严格按照顺序）：

一个 BOM 标记，且这个 BOM 标记必须为 U+FEFF。

- 0-n 个空格或注释。
- DOCTYPE 声明。
- 0-n 个空格或注释。
- 一个 HTML 元素。
- 0-n 个空格或注释。

这里存在着一些和 HTML4 的不同，一个 HTML4 的最简源码文件是这样的：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD
HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<title> 这里是标题 </title>
```

两者的区别是显而易见的：

HTML5 把 DOCTYPE 修改为更简单的 <!DOCTYPE html>，这个已经众所周知了。在 HTML4 中多了一个 <title> 标签。

精简语句吧，让你的MySQL更高效

译者 / 核子可乐

通过几条简单的规则，我们可以方便地提高MySQL 服务器的扩展性。以下是实例。

“改进 SQL 语句最有效的方法是进行精简”

精简 SQL 语句的方法林林总总，但在列举由客户处观察得出的典型范例之前，请允许我先对提高扩展性的基本前提条件做出解释。

MySQL 的核心只允许在给定的时间段（例如每秒）中借由物理过程来运行一定数量的 SQL 语句。无论你的计算机有多么强力，这一物理过程始终存在运算上限。如果你能够将 SQL 语句中那些不具备关键性或必要性的部分精简掉，那么与此同时，真正重要的 SQL 语句也将自动得到优先处理。当然这也将带来其它一些连锁反应，但只是简单数学范畴内的小问题。总之，要运行更多 SQL 指令，首先对你的指令进行精简。

在此我们列举一个简单的例子，通过 mk-query-digest 工具对 TCP/IP 数据包进行分析并输出结果。

```
# Rank Query ID
Response time Calls R/Call Apdx V/M Item # ==== ================== ============= ===== ====== ==== ===== ==========
# 1 0xD631CB919867DB50 0.0436 47.3% 92 0.0005 1.00 0.00 SELECT TTDOD
# 2 0x04FE01C5B31FD305 0.0258 27.9% 329 0.0001 1.00 0.00 ADMIN PING
# 3 0x93321857BCD8E771 0.0229 24.8% 36 0.0006 1.00 0.00 SELECT TTD
```

其中存在很多问题，包括 SQL 的一次一行 (RAT) 特性，不过在这里我们暂不讨论 ping 过多的问题。首先让我们看看第一个语句。

```
SELECT `Date` FROM TTDOD WHERE ID = 9999;
```

表面上看这个查询指令已经够简洁了，但让我们再看看列表。

```
mysql> select count(*) from TTDOD;
+----------+ | count(*) | +----------+
| 0 | +----------+
```

在这种情况下，因为当前列表是空的，所以查询指令将不会返回任何内容。当然这一点在未来可能会发生变化，但就目前来看这更多的是一种在简单数据管理中的异常处理状态，因为该列表中很少会存在内容。而建立一种有针对性的解决方案来通知该应用程序，可以完全避免这类不必要的查询行为。

以上只是个运行时间不足 2 秒的参考实例，而清除第一个查询指令也已经使整体查询时耗降低了 20%。不管这仅仅是个典型的载入过程抑或是批处理中的并行载入过程，原理都是共通的。而且毫无疑问，在接下来的查询指令中，我们的精简工作还大有可为。

原文链接：

http://database.51cto.com/art/201103/252347.htm

编者按

本文为一名开发者在花了一段时间了解新的 Android 3.0 SDK、工具和设备后从开发人员的角度对 Android 3.0 SDK 的优点与不足进行的评判。

开发者评Android 3.0 SDK优缺点

数周前，开发人员收到了 Android 平台的下一个版本蜂巢（Honeycomb，Android 3.0）的预览版，大约 1 个月后，Android 3.0 SDK 发布了最终版，紧接着就出现了第一个 Android 3.0 设备：摩托罗拉 Xoom 平板电脑。我们已经花了一段时间了解新的 SDK、工具和设备，下面是这个平台更新和变化的第一印象，我们会涉及到用户和开发人员的内容，但始终从开发人员的角度来评论。

Android 3.0 让我们兴奋的特性和 API

Android 3.0 提供了各种新的特性和 API，我们特别兴奋的是，它增加了受支持应用程序类型。

Android Fragment

Fragment API 使开发人员更容易动态创建用户界面，更有效地利用屏幕，虽然最初主要是想将其作为平板电脑 API，Fragment API 可用于所有用户界面，简化设计，在不同方向和屏幕大小之间共享更多的代码和布局，使其成为面向不同设备类型必需的 API。

Android Loader

开发人员经常处理的一个问题是在一个 Activity 内下载并显示数据，然后当配置变化时管理这些数据的采集过程，蜂巢引入了 Loader 的概念来解决这个问题，通过异步方式处理数据的提取，以及自动重新连接到数据，避免了重复查询，Loader API 改善了数据在屏幕上显示的性能。

开发者评 Android 3.0 SDK 优缺点 II

Android 活动栏

谷歌在去年的 Google I/O 2010 大会上引入了活动栏(Action Bar)的概念,多个一流的 Android 应用程序包含了活动栏的概念,提供了这种新的用户界面的示范,现在,活动栏的概念已经内置到平台中,并添加了多个新特性,包括下拉菜单,即时访问选项菜单项,标签管理等,旨在为应用程序营造更多的一致性和更简单的导航功能。(拓展阅读:在 Google I/O 2011 大会上谷歌将重点讲解 Android 3.0)

RenderScript

RenderScript是一个承诺提供用C语言编码,可在 CPU 或 GPU 上执行的系统,兼容多种 CPU 和 GPU 类型,它使我想起 OpenCL 和 CUDA,在高性能计算环境中,GPU 的加速效果往往比 CPU 更明显,虽然 GPU 常用于图像和游戏编程,但它的计算能力也可在其它领域大放异彩。

Android 3.0 工具更新

伴随 Android 3.0 的发布,许多工具也进行了更新,图形布局设计器(Graphical Layout)再次做了大幅改进,现在它能更精确地显示在多种 Android 设备,多种屏幕和多个 SDK 版本上的布局预览效果,不用重新编辑 XML 就可实现精确的 Widget 拖放,当然,它仍然不完美,我们发现仍然需要调整 XML 以更好地控制用户界面。

模拟器引入了快照的功能,快照允许模拟器保存某个时刻的状态,以便随后重新载入,节省启动时间。

保持向后兼容的静态库

也许你现在在思考一个问题:这些更新的确很有诱惑力,但现有设备能运行蜂巢吗?是的,不用担心这个问题! Android 团队已经发布了一个静态库,包含了最受欢迎的蜂巢特性,从 Android 1.6 到 Android 2.3.3 都可以使用它们,这个库中最主要的两个特性包含在 Fragment API 和 Loader API 中,就我个人而言,我希望 Action Bar 从一开始就包含进去。

蜂巢的一些遗憾

尽管蜂巢已经很酷了,但它也不是完美无瑕的,这个版本中有些改变不是我们希望看到的,我一直在想谷歌为什么要这么做。

Android 3.0 SDK 模拟器性能

Android 模拟器性能一直以来都备受诟病,其中最让人不爽的就是模拟器的分辨率又提高了,在 Android 3.0 SDK 中内置的模拟器分辨率上升到了 1280x800,导致的结果是,即使我们开发用计算机速度很快,在模拟器中运行应用程序时也只能勉强看到程序的样子,但这也需要一定的耐性的,更别说交互式体验了。我们是开发人员,我们没有耐性这样等一个工具慢慢执行,迄今为止,我们大多数时候都是直接在 Android 设备上调试应用程序的。Android 团队已经意识到这个问题,并表示正努力解决这个问题。我们希望他们尽快解决模拟器的性能问题。

被抛弃的专用按钮

多年以来,开发人员和用户都习惯了 Android 设备上 4 个主要专用按钮:后退,上下文菜单,搜索和主页,现在这些专用按钮消失了,后退和主页被屏幕上的虚拟按钮取代,上下文菜单被活动栏菜单取代,但在应用程序底部显示了上下文菜单,

开发者评 Android 3.0 SDK 优缺点 III

被活动栏菜单取代，但在应用程序底部显示了上下文菜单，搜索按钮也不见了，搜索现在被安排到活动栏中去了，从用户界面设计角度来讲，给开发人员和用户带来的变化太大了。

但从长远来看，这些变化仍然是积极的，按钮现在位于同一个位置，不用再考虑设备和屏幕分辨率的差异，这些变化迫使开发人员和用户重新培养自己的习惯，开发人员必须同时兼顾所有旧设备（有这些专用按钮）和新设备，需要很长一段时间过渡。

Android 3.0 失踪的特性和应用程序

尽管 Android 开发团队在博客上有承诺，但我们从来不期望完美的向前兼容，不过这次还是让我们有点失望，一些特性不见了，例如，Android Market 发生了变化，不再提供应用程序的评论，开发人员不能获得用户的反馈，这真的是不可理喻。

部分平台级用户特性也发生了变化，如用户不能创建文件夹来组织应用程序，不管是从用户还是从开发人员的角度来看，这都是不可接受的。

Android 成长的烦恼

即使谷歌自己的应用程序也受到了蜂巢升级的影响，例如，Google Voice 不能工作，升级到蜂巢的设备，在 Android Market 中根本看不到 Google Voice，有些极客尝试独立安装，发现程序一运行就会崩溃，如果谷歌自己的程序不做修改都不能工作，那说自己可以 100% 向前兼容不是掌自己的嘴吗？普通开发人员还有底气说自己的程序一定可以向前兼容吗？

平台的不稳定对用户或开发人员来说不是一件好事，而媒体是最喜欢报道负面消息的，我们不禁要问："蜂巢（和 Xoom）急着进入市场是为了抢得竞争先机吗？（你应该懂我在说什么）"

小结

总的说来，蜂巢带来了许多令人兴奋的新特性，在许多方面改善了 Android 平台，这些改进也带来了一些变化，开发人员和用户的习惯需要一段时间调整和适应，不是每个人都喜欢或欣赏这些变化，但从长远来看，这些变化有助于 Android 未来的成功，有助于继续蚕食市场份额，迫使其它竞争平台提高注意力，只有竞争才能推动技术进步，最终受益的还是广大的用户。（译者 / 黄永兵）

http://mobile.51cto.com/hot-250081.htm

原 文 名：Android 3.0 Honeycomb SDK: The Good, the Bad and the Missing

作者：Shane Conder

如何安装 Android 插件 ADT

下载 ADT 后，修改 Eclipse ADT 参数指向了 Android SDK 目录：

选择 Window > Preferences.. 打开参数面板（在 Mac OS X 上请选择：Eclipse > Preferences ）。

选择 Android 从左侧面板。

对于 Android SDK 位置在主面板，点击 浏览并找到您下载的 SDK 目录。单击 "应用" 后 ，点击 "确定"。如果你没有遇到任何问题，安装完成 .

Windows Phone开发秘籍

.NET/Windows Phone 7 和 Java/Android Api 之间虽有很多不同点，但两者的相似点让你在移植应用程序时不需要费太大的力气。本文将告诉 Android 开发者如何进行 Windows Phone 开发。

和 Android 平台移动应用程序非常类似，Windows Phone 7（WP7）应用程序也是用托管语言编写的，Android 使用 Java，Windows Phone 7 使用 C#，各自都提供了丰富的开发库，Java 和 C# 之间的许多差异都与样式有关，它们都和 C/C++ 有着千丝万缕的联系，因此它俩也有很多相似之处。

.NET/Windows Phone 7 和 Java/Android Api 之间虽有很多不同点，但两者的相似点让你在移植应用程序时不需要费太大的力气。

平台差异

在深入了解转换或创建 Windows Phone 7 应用程序相关的工具和过程之前，我们先简单介绍一下 Windows Phone 7 和 Android 各自使用的术语和技术。Windows Phone 7 第一个重大的不同点是，应用程序是用 C# 编写的托管 .NET 程序集，平台支持两种类型的应用程序：Silverlight 和 XNA 游戏。

大多数 Windows Phone 7 应用程序都是用 Silverlight 创建的，通过常见的控件，如标签、文本框和列表等，提供典型的基于表单的用户交互，Android Layout 和 Activity 与 Silverlight 中的 Page 类似。

Windows Phone 7 还支持 XNA，它允许用户创建 2D 和 3D 游戏，它们分别等同于 Android SurfaceView 和 GLSurfaceView 的 2D 和 3D 模式，但和 Android 不同的是，XNA 游戏使用的是 Direct3D，因此更容易将 PC 和 Xbox 360 游戏移植到手机上，而 Android 使用的是 OpenGL。

页面（Page）和导航（Navigation）

Silverlight Page 实际上是一个 XML 文件，它和 Android Layout 类似，XML 定义一个 Page 时是使用 XAML（可扩展应用程序标记语言）创建的，XAML 和 Android Layout 类似，但它支持更多的功能，XAML 允许开发人员在 Page 内执行操作，包括 Animations 和 Data Binding 等，从此不再需要为这些功能编写代码。

Android Layout 和 Activity 是松耦合的，你需要自己编写代码，以便和 Activity 内的 UI 元素连接起来，对给定的 Page 来说，C# 代码是和它捆绑到一起的，因此不需要编写 UI 附属元素，平台会自动为对象创建相关的“线路”，为 Page 和 UI 创建相关的事件，这样可以预防在 Task onCreate 方法中经常出现的混乱。

这两个平台另一个重要的不同之处是导航，在 Android 中，你可以通过创建一个 Intent 从一个任务切换到另一个任务，它和 Windows Phone 7 中的 Navigation 功能相同，Navigation 允许你在

Windows Phone 开发秘籍 II

Page 之间移动，和 Android Intent 类似，你可以向你要导航的 Page 传递数据，Windows Phone 7 中的 Page 和 ASP.NET 中的 Page 有某些属性是相似的。

将 Layout 转换成 Page

在深入了解代码之前，我们先来看看 Android Layout XML 如何向 Windows Phone 7 XAML Page 转换，在 Windows Phone 7 免费工具箱中，最耀眼的莫过于 XAML 设计工具，Visual Studio 2010 中的 XAML 设计器只适合初级开发人员，它只能对 Page 做一些基本的控制，Expression Blend 则是专业化的工具，需要改造 UI 时，你应该使用它，和 Android Layout 一样，你也可以使用文本编辑器直接编辑 XAML 文件，因为实际上它是一个 XML 文件。

Windows Phone 7 Page 提供了 Android Layout 类似的布局能力，下表列出了它支持的布局属性和 Android 对应的属性。

布局控件

Windows Phone 7	Android
Canvas	AbsoluteLayout
Grid	GridView
ScrollView	ScrollView
StackPanel	LinearLayout

正如你所看到的，Windows Phone 7 提供了和 Android 几乎相同的布局能力，下表列出了 Windows Phone 7 的基本控件和 Android 的同等控件。

请看下面的对比图

基本控件

Windows Phone 7	Android
TextBlock	TextView
TextBox	EditText
Button	Button
CheckBox	CheckBox
RadioButton	RadioButton
Image	ImageView
ProgressBar	ProgressBar
ListBox	ListView
Map	MapView
WebBrowser	WebView

你应该注意到 Windows Phone 7 的基本控件和 Layout 类型能够一一对应，但也有些特殊的 Layout 和控件无法对应起来，其原因在于 XAML 的强大，通过 XAML，我们可以在一个控件中嵌入另一个控件，这意味着你不用创建复杂的控件，这一切都可以在你的应用程序中创建和自定义。

应用程序存储

对大多数移动应用程序而言，数据的存储是一件大事，对 Windows Phone 7 来说，最佳的办法是选择云服务作为数据存储的主要手段，你可能会感到很奇怪，但如果你是为你的网站开发一个前端程序，你就会明白了。如果你不是做的这种开发，你有两种数据存储方法：使用商业云存储服务，如 Windows Azure 云存储，或使用本地 IsolatedStorage 接口，IsolatedStorage 允许你为应用程序存储文件。

Windows Phone 7 应用程序允许你访问本地数据，但它目前并没有提供内置的数据库 API，如 SQLServer 精简版或 SQLite，如果你现在的（下页）

Windows Phone 开发秘籍 III

Android 应用程序使用的是 SQLite 存储数据，你需要提出一个替代方案，在写这篇文章的时候，在 Codeplex 上有几个开源项目为 IsolatedStorage 接口增加了数据库存储功能。

根据你应用程序的数据存储需要，不需要数据库 API，也许就能将数据存储在本地。

例如，如果你在构建一个 RSS/ Podcast 应用程序，你可以把 RSS 种子就保存在 XML 文件中，当你需要访问数据时，你只需要载入数据，使用 LINQ 获取特定的数据即可。

在 RSS 应用程序示例中，我们可以使用 RSS 种子的原始 XML 文件，也可以使用 LINQ to XML 创建 XDocument 存储下面这样的记录：

```
//Create XML  XDocument doc = new XDocument();
doc.Add(new XElement("DataRoot",
new XElement("Record",
new XElement("value","data1")),
new XElement("Record",
new XElement("value","data2"))  ));
```

这段代码创建的 XML 结构如下：

```
<DataRoot>
<Record>
<value>data1value>  Record>  <Record>
<value>data2value>  Record> DataRoot>
```

为了将这个 XML 隔离存储，我们使用 IsolatedStorageFile 和 IsolatedStorageFileStream，如下面的代码：

```
//Save the XML
using (IsolatedStorageFile isf =
IsolatedStorageFile.GetUserStoreForApplication()) {
using (IsolatedStorageFileStream file = isf.OpenFile("data.xml", FileMode.OpenOrCreate)) {
doc.Save(file);
file.Close();  } }
```

正如你所看到的，隔离存储 XML 数据是如此的简单，读取 XML 和使用 LINQ to XML 查询结果的的代码如下：

```
//Load the XML
using (IsolatedStorageFile isf =
IsolatedStorageFile.GetUserStoreForApplication()) {
using (IsolatedStorageFileStream file = isf.OpenFile("data.xml", FileMode.OpenOrCreate)) {
XDocument d = XDocument.Load(file);
var query = from r in d.Root.Elements("Record")
select r;
//Process the list of record  } }
```

虽然上面的例子非常简单，它也说明了即使没有数据库，在手机上存储数据是多么容易，这些代码还可以进一步扩展，存储更复杂的数据。

Windows Phone 7 支持存储应用程序设置和用户偏好设置，在 Android 中，它叫做 Shared Preferences，支持存储 Key/Value 类型的信息，在 Windows Phone 7 中，它叫做 Local Settings

通过 IsolatedStorage 接口可以访问它们，同样，信息访问也是通过 Key/Value 的格式进行的

开发工具

作为一名 Android 开发人员，你可能对开源开发工具情有独钟，使用 Eclipse 作为 Android 应用程序开发 IDE 无疑是大多数人的选择，但 Android Development Toolkit（ADT）缺乏许多必要的工具，如适合的布局工具，虽然 ADT 带有布局工具，但不够强大，很多时候，我们不得不手写 XML。

相比之下，微软平台可供选择的工具就要多得多，现在在 create.msdn.com 上提供的工具有：

◆ Microsoft Visual Studio 2010 Express for Phone

◆ Microsoft Expression Blend for Phone

◆ Microsoft XNA Game Studio for Phone

这三个工具允许你免费使用，Microsoft Visual Studio 2010 Express for Phone 是主要的 Windows Phone 7 应用程序集成开发环境，它包括了所有必需的工具，如 Page 布局工具，C# 编译器，Windows Phone 7 模拟器等。

Microsoft Expression Blend for Phone 主要是针对专业设计人员的，设计人员可以使用它从零开始创建 Page。

Microsoft XNA Game Studio for Phone 则是为 2D 和 3D 游戏开发准备的，它包括了 XNA 框架，以及处理游戏音效和图像需要的工具。

小结

Windows Phone 7 给智能手机操作系统市场注入了新的活力，它基于成熟的技术，如 Silverlight，WPF，C# 等，正如本文介绍的，Android 和 Windows Phone 7 之间的差异主要集中在表面上，Android 开发人员要转向 Windows Phone 7 开发并不是什么难事。

51CTO 题外话

前段日子诺基亚与微软结盟，未来诺基亚的主推智能手机平台将为 Windows Phone 平台，对诺基亚和微软来讲都将是一个双赢的局面，而对 Windows Phone 平台的开发者来讲有诺基亚这样的手机厂商支持，开发 Windows Phone 应用将更加“钱途”。到目前为止，Android 平台比 Windows Phone 平台的开发者要多得多，希望本文能够给 Android 开发者一个启迪。（译者 / 黄永兵）

原文名：Windows Phone 7 Development for Android Developers

http://mobile.51cto.com/hot-245630.htm

为迎接国际妇女节，向刚走上开发编程和已经编码多年的铿锵玫瑰——IT 技术女性致敬，51CTO 开发频道巨献：资深 IT 技术美女职业分享　破茧化蝶展翼飞！

想查看更多内容，请访问 51CTO 美女节专题，链接：

http://developer.51cto.com/art/201103/247176.htm

谈到 IT 技术的大牛，必然是男性居多，但我们别忘了人类还有一半是女性。其实在 IT 技术领域一直有女性的身影存在，比如第一位程序员是女性。今天我们要细数 IT 技术史上最重要的十五位女性。

用关键词“women in technology”做一个快速的 Google 新闻搜索，结果肯定很令人遗憾，从事技术工作的女性少之又少（也许你只能搜索到 White Town 这张专辑）。去年，NYT 和 WSJ 都发布了这方面的文章，而且前后差不了几周，WSJ 的标题是《Addressing the Lack of Women Leading Tech Start-Ups》，NYT 的文章是用这句话来开头的：真是悲哀啊！从事技术工作的女性在哪里啊？

Google 前美女工程师（图）：软件开发很酷

http://developer.51cto.com/art/201102/246003.htm

Jean Hsu（音译，苏珍妮）曾在 Intel、Google 等公司任职，目前在创业公司做 Web 应用开发工作。曾呼吁更多的女性朋友加入软件开发的大军，这是一件很酷的事情。

专访 51CTO 美女程序员

http://developer.51cto.com/art/201103/247194.htm

哈哈，告诉大家一个秘密：我身边就有个美女程序员：高晶，51CTO 技术部的软件开发工程师，下面是高晶关于她个人对女性程序员这个职业的理解的一段采访。